

OPTIMAL CONTOURS FOR HIGH-ORDER DERIVATIVES

FOLKMAR BORNEMANN AND GEORG WECHSLBERGER

ABSTRACT. As a model of more general contour integration problems we consider the numerical calculation of high-order derivatives of holomorphic functions using Cauchy's integral formula. Bornemann (2011) showed that the condition number of the Cauchy integral strongly depends on the chosen contour and solved the problem of minimizing the condition number for circular contours. In this paper we minimize the condition number within the class of rectangular paths of step size h using Provan's algorithm for finding a *shortest enclosing walk* in weighted graphs embedded in the plane. Numerical examples show that optimal rectangular paths yield small condition numbers even in those cases where circular contours are known to be of not much use, such as for functions with branch-cut singularities.

1. INTRODUCTION

To escape from the ill-conditioning of difference schemes for the numerical calculation of high-order derivatives, numerical quadrature applied to Cauchy's integral formula has on various occasions been suggested as a remedy (for a survey of the literature, see Bornemann 2011). To be specific, we consider a function f that is holomorphic on a complex domain $D \ni 0$; Cauchy's formula gives¹

$$f^{(n)}(0) = \frac{n!}{2\pi i} \int_{\Gamma} z^{-n-1} f(z) dz \quad (1)$$

for each cycle $\Gamma \subset D$ that has winding number $\text{ind}(\Gamma; 0) = 1$. If Γ is not carefully chosen, however, the integrand tend to oscillate at a frequency of order $O(n^{-1})$ with very large amplitude (Bornemann 2011, Fig. 4). Hence, in general, there is much cancelation in the evaluation of the integral and ill-conditioning returns through the backdoor. The condition number of the integral² is (Deuhlhard and Hohmann 2003, Lemma 9.1)

$$\kappa(\Gamma, n) = \frac{\int_{\Gamma} |z|^{-n-1} |f(z)| |dz|}{\left| \int_{\Gamma} z^{-n-1} f(z) dz \right|}$$

and Γ should be chosen as to make that number as small as possible. Since the denominator is, by Cauchy's theorem, independent of Γ , we have to minimize

$$d(\Gamma) = \int_{\Gamma} |z|^{-n-1} |f(z)| |dz|. \quad (2)$$

2010 *Mathematics Subject Classification.* 65E05, 65D25; 68R10, 05C38.

¹Without loss of generality we evaluate derivatives at $z = 0$.

²Given an accurate and stable quadrature method, that condition number actually yields, by

loss of significant digits $\approx \log_{10} \kappa(\Gamma, n)$,

an estimate of the error caused by round-off in the last significant digit of the data (i.e., the function f).

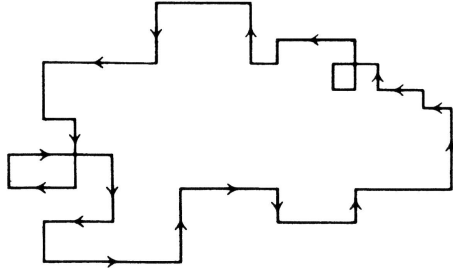


FIGURE 1. Rectangular path of step size h (taken from Lang 1999, Fig. IV.13).

Bornemann (2011) considered circular contours of radius r ; he found that there is a unique $r_* = r(n)$ solving the minimization problem and that there are different scenarios for the corresponding condition number $\kappa_*(n)$ as $n \rightarrow \infty$:

- $\kappa_*(n) \rightarrow \infty$, if f is in the Hardy space H^1 ;
- $\limsup_{n \rightarrow \infty} \kappa_*(n) \leq M$, if f is an entire function of completely regular growth which satisfies a non-resonance condition of the zeros and whose Phragmén–Lindelöf indicator possesses M maxima (a small integer).

Hence, though those (and similar) results basically solve the problem of choosing proper contours for entire functions, much better contours have to be found for the class H^1 . Moreover, the restriction to circles lacks an algorithmic flavor that would point to more general problems depending on the choice of contours, such as the numerical solution of highly-oscillatory Riemann–Hilbert problems (Olver 2011).

In this paper, we solve the contour optimization problem within the more general class of rectangular pathes of step size h (see Fig. 1) as they are known from Artin’s proof of the general, homological version of Cauchy’s integral theorem (Lang 1999, IV.3). Such paths are composed from horizontal and vertical edges taken from a (bounded) rectangular grid $\Omega_h \subset D$ of step size h . Now, the weight function (2), being *additive* on the abelian group of path chains, turns the grid Ω_h into an edge-weighted graph such that each optimal rectangular path W_* becomes a *shortest enclosing walk* (SEW); “enclosing” because we have to match the winding number condition $\text{ind}(W_*; 0) = 1$. An efficient solution of the SEW problem for embedded graphs was found by Provan (1989) and serves as a starting point for our work.

Outline of the Paper. In Section 2 we discuss general embedded graphs in which an optimal contour is to be searched for; we discuss the problem of finding a shortest enclosing walk and recall Provan’s algorithm. In Section 3 we discuss some implementation details and optimizations for the problem at hand. Finally, in Section 4 we give some numerical examples and compare with the optimal circles obtained by Bornemann (2011).

2. CONTOUR GRAPHS AND SHORTEST ENCLOSING WALKS

By generalizing the grid Ω_h , we consider a finite graph $G = (V, E)$ built from vertices $V \subset D$ and edges E that are smooth curves connecting the vertices *within* the domain D . We write uv for the edge connecting the vertices u and v ; by (2), its

weight is defined as

$$d(uv) = \int_{uv} |z|^{-n-1} |f(z)| d|z|. \quad (3)$$

A *walk* W in the graph G is a *closed* path built from a sequence of adjacent edges, written as (where $\dot{+}$ denotes joining of paths)

$$W = v_1v_2 \dot{+} v_2v_3 \dot{+} \cdots \dot{+} v_mv_1;$$

it is called *enclosing* the obstacle 0 if the winding number is $\text{ind}(W; 0) = 1$. The set of all possible enclosing walks is denoted by Π . As discussed in §1, the condition number is optimized by the shortest enclosing walk (not necessarily unique)

$$W_* = \underset{W \in \Pi}{\text{argmin}} d(W)$$

where, with $W = v_1v_2 \dot{+} v_2v_3 \dot{+} \cdots \dot{+} v_mv_1$ and $v_{m+1} = v_1$, the *total weight* is

$$d(W) = \sum_{j=1}^m d(v_jv_{j+1}).$$

The problem of finding such a SEW was solved by Provan (1989): The idea is that with $\mathcal{P}_{u,v}$ denoting a shortest path between u and v , any shortest enclosing walk $W_* = w_1w_2 \dot{+} w_2w_3 \dot{+} \cdots \dot{+} w_mw_1$ can be cast in the form (Provan 1989, Thm. 1)

$$W_* = \mathcal{P}_{w_1, w_j} \dot{+} w_jw_{j+1} \dot{+} \mathcal{P}_{w_{j+1}, w_1}$$

for at least one j . Hence, any shortest enclosing walk W_* is already specified by one of its vertices and one of its edges; therefore

$$W_* \in \tilde{\Pi} = \{\mathcal{P}_{u,v} \dot{+} vw \dot{+} \mathcal{P}_{w,u} : u \in V, vw \in E\}.$$

Provan's algorithm finds W_* by, first, building the finite set $\tilde{\Pi}$; second, by removing all walks from it that do not enclose $z = 0$; and third, by selecting a walk from the remaining candidates that has the lowest total weight. Using Fredman and Tarjan's (1987) implementation of Dijkstra's algorithm to compute the shortest paths $\mathcal{P}_{u,v}$, the run time of the algorithm is known to be (Provan 1989, Corollary 2)

$$O(|V| |E| + |V|^2 \log |V|).$$

3. IMPLEMENTATION DETAILS

3.1. Edge Weight Calculation. Using the edge weights $d(uv)$ requires to approximate the integral in (3). Since not much accuracy is needed, a simple trapezoidal rule with two nodes is generally sufficient:

$$\begin{aligned} d(uv) &= \int_{uv} |z|^{-(n+1)} |f(z)| d|z| \\ &\approx \frac{|u-v|}{2} (d(u) + d(v)) = \tilde{d}(uv) \end{aligned}$$

with the *vertex weight*

$$d(z) = |z|^{-(n+1)} |f(z)|. \quad (4)$$

Although $\tilde{d}(uv)$ will typically have an accuracy of not more than just a few bits, we have not encountered a single case, in which a more accurate computation of the weights would have resulted in a different SEW W_* .

3.2. Reducing the size of $\tilde{\Gamma}$. As described in Section 2, Provan’s algorithm starts by building a walk for every pair $(v, e) \in V \times E$ and then proceeds by selecting the best enclosing one. A simple heuristics, which worked well for all our test cases, helps to considerably reduce the number of walks to be processed: Let

$$v_* = \operatorname{argmin}_{v \in V} d(v)$$

and define W_{v_*} as a SEW subject to the constraint

$$W_{v_*} \in \tilde{\Gamma}_{v_*} = \{\mathcal{P}_{v_*,u} \dot{+} uw \dot{+} \mathcal{P}_{w,v_*} : uw \in E\}.$$

Obviously W_* and W_{v_*} do not need to agree in general, as v_* does not have to be traversed by W_* . However, since v_* is the vertex with lowest weight, both walks differ mainly in a region that has no, or very minor, influence on the total weight and, consequently, also no significant influence on the condition number. Actually, W_* and W_{v_*} yielded precisely the same total weight for all functions that we have studied. Using that heuristics, the run time of Provan’s algorithm improves to

$$O(|E| + |V| \log |V|),$$

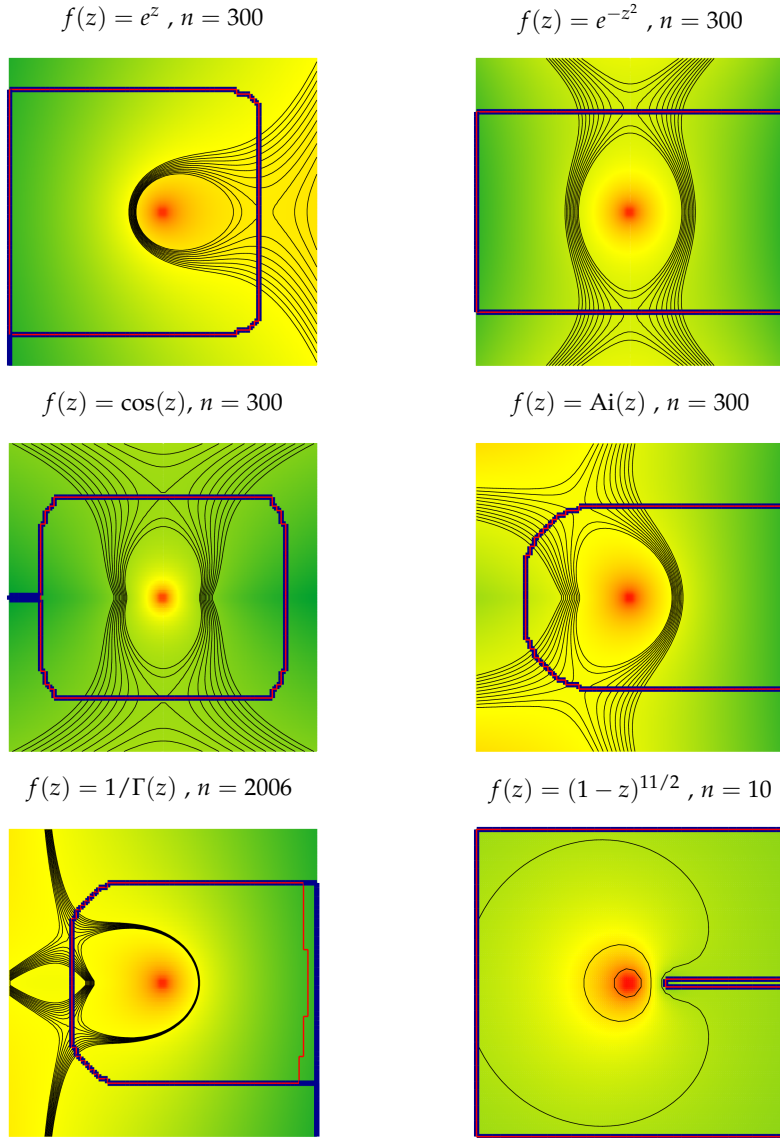
because its main part reduces to applying Dijkstra’s shortest path algorithm just once. Fig. 2 compares W_* and W_{v_*} for a few examples.

3.3. Size of the Grid Domain. We restrict ourselves to graphs given by finite square grids of step size h , centered at $z = 0$ —with all vertices and edges removed that do not belong to the domain D . Since Provan’s algorithm just requires an embedded but not a planar graph, we may add the diagonals of the grid cells as further edges to the graph. These additional edges are advantageous, e.g., in all those cases which sensitively depend on properly matching the direction of steepest descent in the saddle points of $d(z)$ (Bornemann 2011, §9).

The side length l of the square domain (that is to be subdivided by the grid) has to be chosen large enough to contain a SEW that would approximate an optimal general integration contour. For example, if f is entire, we choose $l > 2r_*$, where r_* is the radius of the optimal circular contour given in Bornemann (2011). In other cases we employ a simple search for a suitable value of l by calculating W_* for increasing values of l until $d(W_*)$ does not substantially decrease anymore. During this search each grid uses just a fixed number of vertices.

3.4. Multilevel Refinement of the SEW. Choosing a proper value of h is not straightforward since we would like to balance a good approximation of a generally optimal integration contour with a reasonable amount of computing time. In principle, we construct a sequence of SEWs for smaller and smaller values of h until the weight of W_* does not substantially decrease anymore. To avoid an unduly amount of computational work, we do not refine the grid everywhere but use an adaptive refinement by confining it to a tubular neighborhood of the currently given SEW W_* (see Fig. 3):

- 1: calculate W_* within an initial grid;
- 2: subdivide each rectangle adjacent to W_* into 4 rectangles;
- 3: remove all other rectangles;
- 4: calculate W_* in the newly created graph.



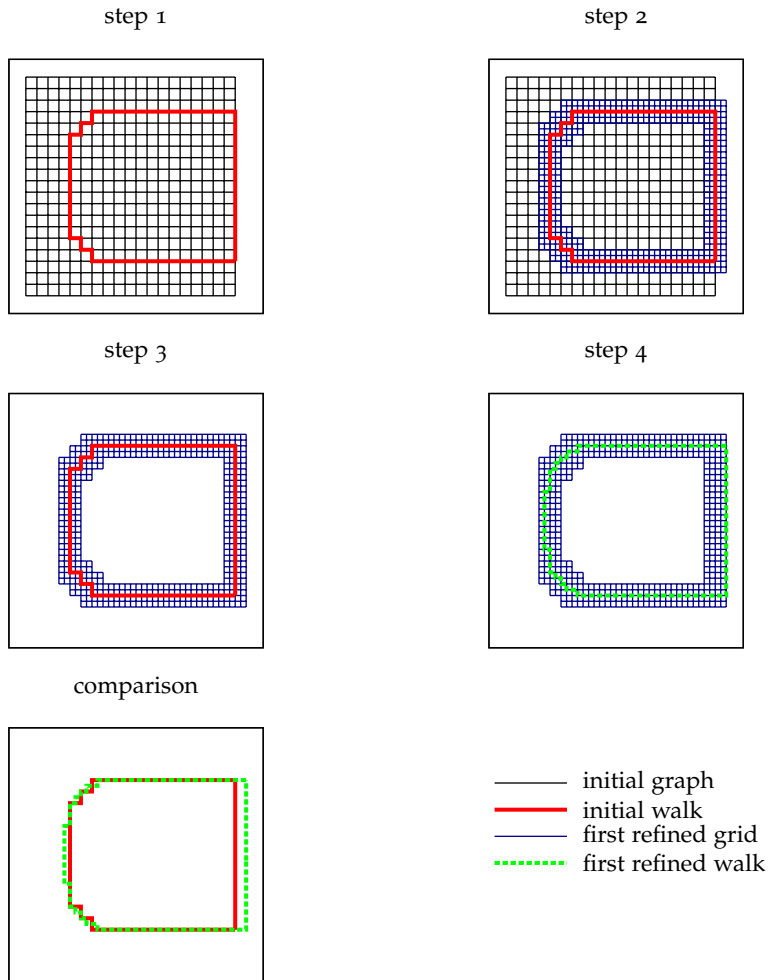


FIGURE 3. Multilevel refinement of W_* ($f(z) = 1/\Gamma(z)$, $n = 2006$)

As long as the total weight of W_* decreases substantially, steps 2 to 4 are repeated. It is even possible to optimize that process further by not subdividing rectangles that just contain vertices or edges of W_* having weights below a certain threshold.

3.5. Quadrature Rule for the Cauchy Integral. Finally, after calculation of the SEW $\Gamma = W_*$, the Cauchy integral (1) has to be evaluated by some accurate numerical quadrature. We decompose Γ into maximally straight line segments, each of which can be a collection of many edges. On each of those line segments we employ Clenshaw–Curtis quadrature in Chebyshev–Lobatto points. Additionally we neglect segments with a weight smaller than 10^{-24} times the maximum weight of an edge of Γ , since such segments will not contribute to the result within

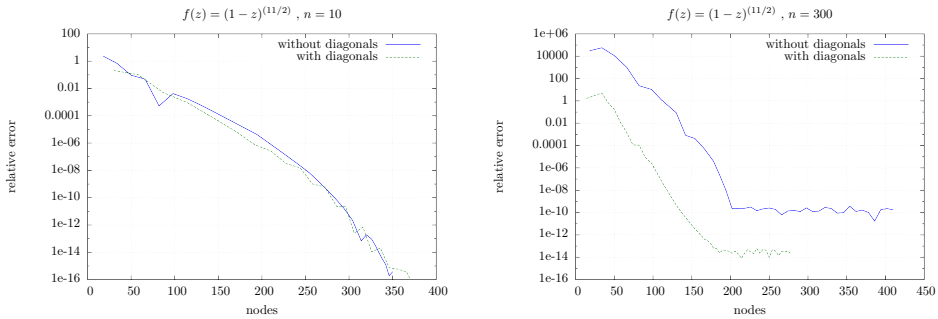


FIGURE 4. Illustration of the spectral accuracy of piecewise Clenshaw–Curtis quadrature on SEW contours for a function with a branch-cut singularity. For larger n , we observe a significant improvement by adding diagonals to the grid. We get to machine precision for $n = 10$ and loose about two digits for $n = 300$. Note that for optimized circular contours the loss would have been about 6 digits for $n = 10$ and about 15 digits for $n = 300$ (cf. Bornemann 2011, Thm. 4.7).

TABLE 1. Condition numbers for some $f(z)$: r_* are the optimal radii given in Bornemann (2011); W_* was calculated in all cases (except the last one) on a 70×70 -grid with $l = 3r_*$ (in the last case l was found by the method of §3.3). For $1/\Gamma(z)$, the specific orders of differentiation $n = 2006$ and $n = 10935$ are two of those very rare orders that give exceptionally large condition numbers (cf. Bornemann 2011, Table 5).

$f(z)$	n	$\kappa(W_*, n)$	$\kappa(C_{r_*}, n)$
e^z	300	2.0	1.0
$\cos z$	300	1.1	1.0
e^{-z^2}	300	1.2	1.0
$\text{Ai}(z)$	300	1.6	1.2
$1/\Gamma(z)$	300	2.2	1.6
$1/\Gamma(z)$	2006	$7.8 \cdot 10^4$	$4.7 \cdot 10^4$
$1/\Gamma(z)$	10935	$1.6 \cdot 10^5$	$1.4 \cdot 10^5$
$(1-z)^{11/2}$	10	7.7	$5.0 \cdot 10^5$

machine precision. This way we not only get spectral accuracy but also, in many cases, less nodes than the vanilla version of trapezoidal sums on a circular contour would need: Fig. 4 shows an example with the order $n = 300$ of differentiation but accurate solutions using just about 200 nodes which is well below what the sampling condition would require for circular contours (Bornemann 2011, §2.1). Of course, trapezoidal sums would also benefit from some recursive device that helps to neglect those nodes which do not contribute to the numerical result.

4. NUMERICAL RESULTS

Table 1 displays condition numbers of SEWs W_* on rectangular grids as compared to the optimal circles C_{r_*} for a couple of functions; Fig. 5 shows some of the

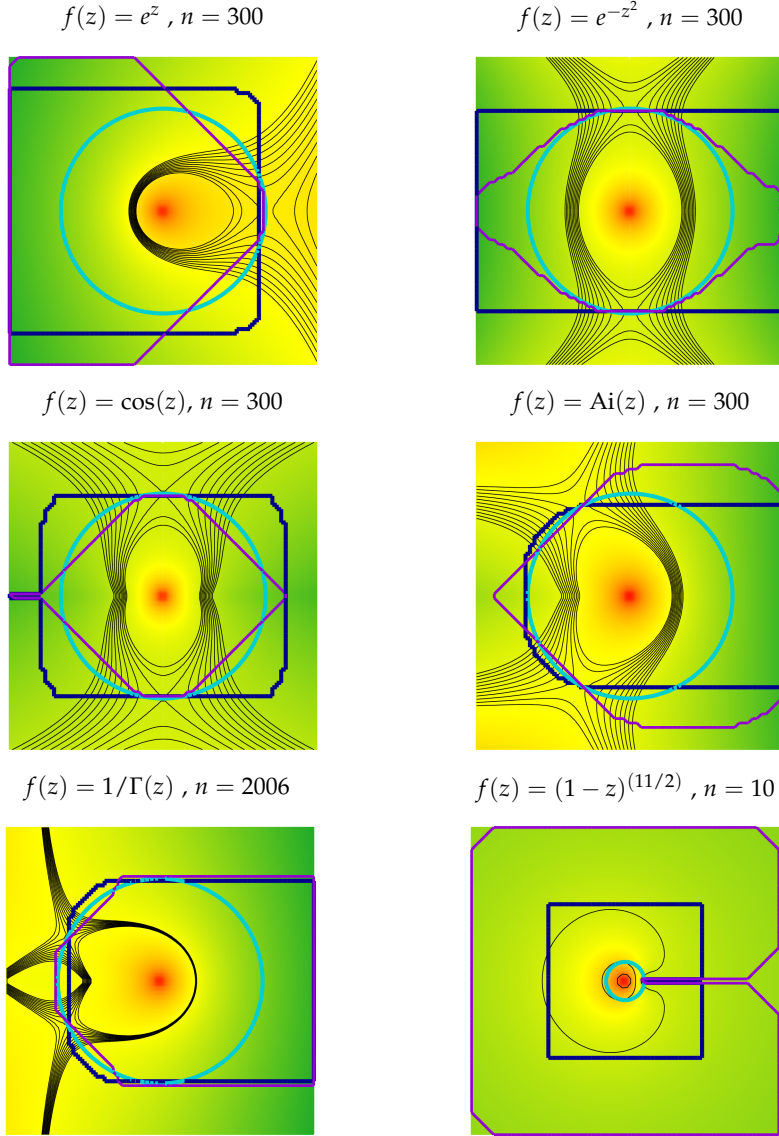


FIGURE 5. W_{v_*} (admissible paths: blue = rectangular, magenta = rectangular with diagonals enabled) vs. C_{r_*} (cyan) for some examples of Table 1: the color coding shows the size of $\log d(z)$; with red for large values and green for small values. The thin black lines are level curves of $\log d(z)$; the smallest level shown is the threshold, below of which the edges of W_{v_*} do not contribute to the first significant digits of the total weight.

corresponding contours. For entire f we observe that W_* , like the optimal circle C_{r_*} , automatically traverses the saddle points of $d(z)$. It was shown in Bornemann (2011, Thm. 10.1) that, for such f , the major contribution of the condition number

comes from these saddle points and that circles are (asymptotically, as $n \rightarrow \infty$) paths of steepest decent. Since W_* can cross a saddle point only in a horizontal, vertical, or (if enabled) diagonal direction, slightly larger condition numbers have to be expected. Nevertheless, the order of magnitude is precisely matched. This match holds in cases where circles give a best possible condition number of approximately 1, as well as in cases with exceptionally large condition numbers, such as for $f(z) = 1/\Gamma(z)$ with the exceptional orders of differentiation $n = 2006$ and $n = 10935$ (cf. Bornemann 2011, §10.4).

For some non-entire f , however, optimized circles can be far from optimal in general: Bornemann (2011, Thm. 4.7) shows that the optimized circle C_{r_*} for functions f from the Hardy space H^1 with boundary values in $C^{k,\alpha}$ yields a lower condition number bound of the form

$$\kappa(C_{r_*}, n) \geq cn^{k+\alpha};$$

for instance, $f(z) = (1 - z)^{11/2}$ gives $\kappa(C_{r_*}, n) \sim 0.16059 \cdot n^{13/2}$; the principal branch of that f has a branch cut at $(1, \infty)$ and W_* gives significantly better condition numbers than C_{r_*} by automatically following the cut.

Conclusion. We observe that contours optimized in the class of rectangular paths are a flexible tool covering different classes of functions in a completely algorithmic fashion; no deep theory is needed to let the computation run (the theory is only required to explain large condition numbers if they cannot be avoided, such as for the entire function $1/\Gamma$ and certain orders n of differentiation).

REFERENCES

- Bornemann, F.: 2011, Accuracy and stability of computing high-order derivatives of analytic functions by Cauchy integrals, *Found. Comput. Math.* **11**, 1–63.
- Deuffhard, P. and Hohmann, A.: 2003, *Numerical analysis in modern scientific computing*, second edn, Springer-Verlag, New York.
- Fredman, M. L. and Tarjan, R. E.: 1987, Fibonacci heaps and their uses in improved network optimization algorithms, *J. Assoc. Comput. Mach.* **34**, 596–615.
- Lang, S.: 1999, *Complex analysis*, fourth edn, Springer-Verlag, New York.
- Olver, S.: 2011, Numerical solution of Riemann–Hilbert problems: Painlevé II, *Found. Comput. Math.* **11**, 153–179.
- Provan, J. S.: 1989, Shortest enclosing walks and cycles in embedded graphs, *Inform. Process. Lett.* **30**, 119–125.

ZENTRUM MATHEMATIK – M3, TECHNISCHE UNIVERSITÄT MÜNCHEN, 80290 MÜNCHEN, GERMANY
E-mail address: bornemann@tum.de; wechslbe@ma.tum.de